**Jamieson T. Olsen**

# Engineering Note

**Date:** 28 January 2000
**Rev Date:** 6 January 2002

**Project:** Digital Front End Board
**Doc. No:** 2000-01-28a

**Subject:** DFE and Mixer Board Registers

This document does NOT describe high level DFEC commands. This document describes the register level transactions that occur between the DFEC and a DFE board or MIXER board across the crate backplane.

## *Base Addresses*

Each slot in a the crate is assigned a unique 5-bit number. Slot 1 is reserved for the DFEC and has no address. Slot 2 is assigned address 00010; slot 3 is 00011, and so on. This base address is hard-wired into the backplane so that no jumpers are required. The 5 base address bits are mapped into the 16-bit address space as bits [12..8]. Therefore the base address for each slot is defined as:

| slot | base address | slot | base address |
|---|---|---|---|
| 1 | n/a | 12 | 0x0C00 |
| 2 | 0x0200 | 13 | 0x0D00 |
| 3 | 0x0300 | 14 | 0x0E00 |
| 4 | 0x0400 | 15 | 0x0F00 |
| 5 | 0x0500 | 16 | 0x1000 |
| 6 | 0x0600 | 17 | 0x1100 |
| 7 | 0x0700 | 18 | 0x1200 |
| 8 | 0x0800 | 19 | 0x1300 |
| 9 | 0x0900 | 20 | 0x1400 |
| 10 | 0x0A00 | 21 | 0x1500 |
| 11 | 0x0B00 | | |

## *Register Map*

The following table defines the 8 **byte-wide** registers for each board.

| | read | write |
|---|---|---|
| base | board status | board control |
| +1 | device status | device control |
| +2 | device config data | device config data |
| +3 | device features* | device features* |

2000-01-28a

* application specific

Boards may use additional registers to support board specific functions and diagnostics. Refer to that board's documenation for information about these addtional registers.

## *Register Definitions*

| BOARD STATUS REGISTER (READ BASE + 0) | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BOARD TYPE CODE | | | | CLOCK SELECT | | RESET | READY |

Bits 7..4 The board type code describes the type of board installed in the slot. The bits are defined in my engineering note 2001-09-25A.

Bits 3..2 Clock Selection bits. The boards can dynamically choose a source for the master board clock. DFE motherboards and Mixer boards have different meanings for these bits:

| clock select | | DFE | Mixer |
|---|---|---|---|
| 0 | 0 | link2 | link0 |
| 0 | 1 | link7 | backplane clock |
| 1 | 0 | xtal | xtal |
| 1 | 1 | xtal | auto |

Bit 1 This bit comes back HIGH if the board is currently in RESET. (Reset does NOT cause devices to lose their configuration.)

Bit 0 The READY bit is set if all devices are configured.

| BOARD CONTROL REGISTER (WRITE BASE + 0) | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| X | | | | CLOCK SELECT | | RESET | X |

Bits 7..4: Don't care. Whatever data is written here is ignored.

Bits 3..2: Clock Selection bits. The value written here determines which clock source will be used as the master board clock.

| clock select | | DFE | Mixer |
|---|---|---|---|
| 0 | 0 | link2 | link0 |
| 0 | 1 | link7 | backplane clock |
| 1 | 0 | xtal | xtal |
| 1 | 1 | xtal | auto |

Bit 1: Writing a 1 here will cause the board to RESET after a 2 second delay. RESET will continue to be asserted as long as this bit is set. Reset does not cause devices to lose their configuration. This bit should be 0 when configuring any devices on the board.

Bit 0: Don't care. Whatever data is written here is ignored.

| DEVICE STATUS REGISTER (READ BASE + 1) | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| DONE | PROG | INIT | REG | DEVICE SELECT | | | |

Bit 7: The DONE bit goes high after the specified device has received all of its configuration data.

Bit 6: Returns the status of the PROGRAM line. If this bit is high, it means that the PROGRAM line is being asserted by the board. In this state the specified device is clearing its configuration memory and preparing to accept configuration data. This bit must be low before configuration data can be sent to the device; it will ignore the configuration data otherwise.

Bit 5: If the INIT bit is low there was a problem during device configuration. If this is the case, assert the program bit (see below) and send the configuration data again.

Bit 4: Returns the status of the linear voltage regulator. If this bit is low, there could be a problem with excessive power consumption on the device – configuration data may be maintained, but error free device operation cannot be guaranteed. This bit is not used on the mixer and it is always zero.

Bits 3..0: Returns the current device number. Up to 16 devices per board are supported. Numbering starts at 0.

| DEVICE CONTROL REGISTER (WRITE BASE + 1) | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| X | PROG | X | X | DEVICE SELECT | | | |

Bit 7: Don’t care. Writes to this bit are ignored.

Bit 6: Assert the device PROGRAM bit. To configure a specific device, first write a 1 here. This will cause the specified device to erase its configuration memory and begin preparing to accept new configuration data. Then write a 0 to this bit. Now the specified device is ready to accept configuration data.

Bits 5..4: Don’t care. Writes to these bits are ignored.

Bit 3..0: Specify the target device. Up to 16 devices per board are supported. The first device is 0. Once the current device is set by writing to this register it remains unchanged until a power cycle occurs (default value is 0). Asserting reset does NOT change the value of this register.

| DEVICE CONFIG DATA REGISTER (WRITE BASE + 2) | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

| DEVICE CONFIG DATA REGISTER (READ BASE + 2) | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

Configuration data bytes are written to this register. Bit 7 is the MSb, Bit 0 the LSb. The board will route the configuration data byte to the current device. (NOTE: to set the current device write to the DEVICE CONTROL REGISTER – see above.)

Reading the DEVICE CONFIG DATA REGISTER simply returns the value last written to this register.

| DEVICE FEATURES REGISTER (WRITE BASE + 3) | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

| DEVICE FEATURES REGISTER (READ BASE + 3) | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

Writing to the DEVICE FEATURES REGISTER allows the user to pass an 8-bit value to the current device. (This only works if the current device was configured with firmware that accepts post-configuation arguements.) The meaning of the data written to this register is user defined and is specific to the board/firmware application. For example, the contents of this register may mean the L3 pipeline depth, the device’s home sector, the firmware revision, etc.

This register can be written at any time, and does NOT require a board reset to cause the contents to be transferred into the current device. Note that the data transfer to the current device is uni-directional: reading this register will return the value that was last written to the register – it is not possible to read data from the device using this register.

## Oddities and Inconsistancies

Different boards treat the DONE and PROG bits differently, and it affects how they are configured and verified.

**DFEA BOARDS**

On DFEA boards (board type code = 0000) each device has its own unique PROG and DONE bit. What that means is that a 0-1-0 transition on device 0's PROG bit does not affect device 1 in any way and vise versa. After device 0 receives its configuration data the DONE bit will go high, regardless of the status of device 1.

**MIXER BOARDS**

Mixer board devices share a common PROG bit. What that means is that if the PROG bit makes a 0-1-0 transition, any devices on the board that were previously programmed will loose their programming. To get around this potential problem, only device 0's PROG will actually control the PROG net on the Mixer board. Toggling the PROG bit on other devices will not affect other devices (but it will allow writes to each device's FIRWARE ID REGIGISTER to succeed when that device's PROG bit is high). Thus, when configuring devices on a mixer board you must start with device 0 and proceed until all devices have been configured.

DONE bits, on the other hand, are separate and unique. After each device receives its configuration data, its DONE bit will go high regardless of the status of the other devices on the mixer board.

**ALL OTHER BOARDS**

Like the Mixer board, the PROG bit is common to all devices. Only device 0 actually controls the PROG net on the board.

Unlike the Mixer board, the DONE bit is common to all devices. Unconfigured devices will pull the DONE bit low until they are configured. Thus, the DONE bit will not go high until all devices on the board have been configured.

### *DFEA Example…*

The task is to configure a board in slot 2 of a crate. This board is a DFE motherboard containing two DFEA daughterboards.

1. Read the BOARD STATUS REGISTER (for this board it's address 0x0200). Examine the upper nibble. It is zero. From this information the you know that the board is a DFE motherboard configured as a DFEA board. From this document you know that it has two devices (0 and 1) that need to be configured. Now examine the lower nibble. It is also zero. This means that the board is not being held in RESET. Good.
2. The next thing to do is select the first device and get it ready to accept configuration data. To clear the configuration memory for device 0 the PROG bit must make a 0-1-0 transition.
   a. Write 0x00 to the DEVICE CONTROL REGISTER.
   b. Write 0x40 to the DEVICE CONTROL REGISTER.
   c. Write 0x00 to the DEVICE CONTROL REGISTER.
3. The current device is now 0. Now read the DEVICE STATUS REGISTER. The lower nibble should be zero because the current device is still 0. Verify that the PROG and DONE bits are both zero.
4. Now write the configuration data bytes to the DEVICE CONFIGURATION REGISTER. The configuration data will be written into the current device, which you verified was 0 in the previous step.
5. After all of the configuration bytes have been written to device 0, check that is was successful. Read the DEVICE STATUS REGISTER. It reads 0x80. The DONE bit is set, which means that device 0 has successfully configured.
6. Now you need to configure the second device (1). It too must have the PROG bit toggled 0-1-0 prior to sending configuration data.
   a. Write 0x01 to the DEVICE CONTROL REGISTER.
   b. Write 0x41 to the DEVICE CONTROL REGISTER.
   c. Write 0x01 to the DEVICE CONTROL REGISTER.
7. The current device is 1. Now read the DEVICE STATUS REGISTER. The lower nibble should be zero because the current device is still 1. Verify that the PROG and DONE bits are both zero.
8. Now write the configuration data bytes to the DEVICE CONFIGURATION REGISTER. The configuration data will be written into the current device, which you verified was 1 in the previous step.
9. After all of the configuration bytes have been written to device 1, check that is was successful. Read the DEVICE STATUS REGISTER. It reads 0x81. The DONE bit is set, which means that device 1 has successfully configured.
10. Since all devices have been configured, the board is ready. Verify this by reading from the BOARD STATUS REGISTER and checking to see that bit 0 is high and the reset bit is low.
11. Now verify the firmware ID's for the devices. The current device is still 1. Set it back to device zero by writing 0x00 to the DEVICE CONTROL REGISTER.

### *Mixer Board Example…*

The task is to configure a mixer board in slot 17 of a crate.

1. Read the BOARD STATUS REGISTER (for this board it's address 0x1100). Examine the upper nibble. It is 0xD. From this information the you know that the board is a Mixer board. From this document you know that it has 16 devices (0-15) that need to be configured. Now examine the lower nibble. It is also zero. This means that the board is not being held in RESET. Good.
2. The next thing to do is select the first device and get it ready to accept configuration data. To clear the configuration memory for device 0 the PROG bit must make a 0-1-0 transition. While PROG is high you should write to the FIRMWARE ID REGISTER. (This is the only time that the firmware register can be written.) So do it like this:
    a. Write 0x00 to the DEVICE CONTROL REGISTER.
    b. Write 0x40 to the DEVICE CONTROL REGISTER.
    c. Write the firmware ID byte (say 0x4A) to the FIRMWARE ID REGISTER.
    d. Write 0x00 to the DEVICE CONTROL REGISTER.
3. The current device is now 0. Now read the DEVICE STATUS REGISTER. The lower nibble should be zero because the current device is still 0. Verify that the PROG and DONE bits are both zero.
4. Now write the configuration data bytes to the DEVICE CONFIGURATION REGISTER. The configuration data will be written into the current device, which you verified was 0 in the previous step.
5. After all of the configuration bytes have been written to device 0, check that is was successful. Read the DEVICE STATUS REGISTER. It reads 0x80. The DONE bit is set, which means that device 0 has successfully configured.
6. Now you need to configure the remaining devices. Repeat steps 2-5 for the rest of the devices. (Remember to change the lower nibble of the byte when writing the DEVICE CONTROL REGISTER to specify the device you're targeting.)
7. Since all devices have been configured, the board is ready. Verify this by reading from the BOARD STATUS REGISTER and checking to see that bit 0 is high and the reset bit is low.
8. Now verify the firmware ID's for the devices. (optional)

## *“Other” Board Example…*

The task is to configure a CTOC board in slot 12 of a crate.

1. Read the BOARD STATUS REGISTER (for this board it’s address 0x0C00). Examine the upper nibble. It is 0x1. From this information the you know that the board is a CTOC board. From this document you know that it has 3 devices (0-2) that need to be configured. Now examine the lower nibble. It is also zero. This means that the board is not being held in RESET. Good.
2. The next thing to do is select the first device and get it ready to accept configuration data. To clear the configuration memory for device 0 the PROG bit must make a 0-1-0 transition. While PROG is high you should write to the FIRMWARE ID REGISTER. (This is the only time that the firmware register can be written.) So do it like this:
    a. Write 0x00 to the DEVICE CONTROL REGISTER.
    b. Write 0x40 to the DEVICE CONTROL REGISTER.
    c. Write the firmware ID byte (say 0x79) to the FIRMWARE ID REGISTER.
    d. Write 0x00 to the DEVICE CONTROL REGISTER.
3. The current device is now 0. Now read the DEVICE STATUS REGISTER. The lower nibble should be zero because the current device is still 0. Verify that the PROG and DONE bits are both zero.
4. Now write the configuration data bytes to the DEVICE CONFIGURATION REGISTER. The configuration data will be written into the current device, which you verified was 0 in the previous step.
5. Don’t check device 0’s DONE bit. (It won’t go high until all devices are configured!)
6. Now you need to configure the remaining devices. Repeat steps 2-5 for the rest of the devices. (Remember to change the lower nibble of the byte when writing the DEVICE CONTROL REGISTER to specify the device you’re targeting.)
7. Since all devices have been configured, the board is ready. Verify this by reading from the BOARD STATUS REGISTER and checking to see that bit 0 is high and the reset bit is low.
8. Now verify the firmware ID’s for the devices. (optional)